Министерство образования и науки РФ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Уральский государственный педагогический университет»
Институт математики, информатики и информационных технологий
Кафедра информатики, информационных технологий
и методики обучения информатике

КРОССПЛАТФОРМЕННЫЙ КОНСТРУКТОР КОМБИНАЦИОННЫХ СХЕМ

Выпускная квалификационная работа
по специальности 0502002.65 – Информатика

Исполнитель: студент группы ИС-41
института ИМиИТ
Солонинина А. О.

Руководитель: к.п.н., доцент кафедры ИКТО
Стариченко Е. Б.

Работа допущена к защите
«_12_»___мая___ 2016 г.
Зав. кафедрой _____

Екатеринбург – 2016

Реферат

Солонинин А. О.. КРОССПЛАТФОРМЕННЫЙ КОНСТРУКТОР КОМБИНАЦИОННЫХ СХЕМ, выпускная квалификационная работа: 46 стр., рис. 40, библ. 29. назв.

Ключевые слова: ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, LINUX, WINDOWS, QT, PYTHON, КОМБИНАЦИОННЫЕ СХЕМЫ.

Объект разработки – способы автоматизации построения и расчета комбинационных схем.

Цель работы – создать программу, работающую в операционной системе на базе Linux, и способную осуществлять расчет таблицы истинности и визуально отображать его для комбинационной схемы, заданной пользователем.

В работе описаны результаты проектирования и реализации программного обеспечения, позволяющей конструировать комбинационные схемы, с последующим подсчетом таблицы истинности.

Оглавление

Введение

При разработке различных цифровых систем проектировщики часто сталкиваются с необходимостью создания комбинационных схем. Кроме комбинационных частей конечных автоматов, комбинационные схемы широко используются в качестве оригинальных функциональных узлов многих устройств, которые мы используем каждый день.

Простейшая цепь, зажигающая свет или схема включения компьютера — построены на одних и тех же законах. Какой бы сложной ни была структура конкретной системы, ее можно изучать и описывать по частям, выявляя в ней сравнительно небольшое число стандартных элементов, составляющих стандартные узлы и блоки.

Описание и работа отдельных элементов основываются на применении формального метода математики к области логики. Подобно тому, как в математике для выражения отношений используется язык формул, в математической логике тот же язык формул используется для выражения логических связей, которые существуют между суждениями, понятиями, высказываниями. Такая тесная связь с математикой обеспечивает простоту и хорошие возможности для автоматизации построения и моделирования подобного рода схем.

При моделировании, особенно сложных схем, трудно обойтись без визуального представления. Наличие визуализации помогает лучше ориентироваться в моделируемой системе, а также предоставляет совсем иной подход к моделированию. Однако мало отображать процессы, пользователю необходимо иметь возможность полностью смоделировать систему, пользуясь наглядным визуальным редактором. Программное обеспечение с данной функциональностью относится к разряду проблемно-ориентированного программного обеспечения. Многие программные продукты привязаны к платформе ОС Windows и в широком доступе такого ПО нет.

Противоречие между необходимостью визуального построения сложных комбинаторных схем и отсутствием подходящего программного обеспечения определяет актуальность нашей работы.

Предметом исследования являются комбинационные схемы.

Объектом исследования – способы автоматизации построения и расчета комбинационных схем.

Цель работы: создать программу, работающую в операционной системе на базе Linux, и способную осуществлять расчет таблицы истинности и визуально отображать его для комбинационной схемы, заданной пользователем.

Задачи:

- Произвести анализ литературы по теме работы.
- Проанализировать и сравнить программы для конструирования комбинационных схем.
- Разработать алгоритм программы, предназначенной для автоматизации построения комбинационной схемы.
- Написать программу, согласно разработанному алгоритму.

1. Основы математического аппарата анализа комбинационных логических схем и программы для их составления

1.1 Комбинационные устройства

Англичанин Чарльз Бэббидж один из первых людей, кто задумался об автоматизации процессов вычисления. В начале XIX века он начал продумывать машину, способную производить расчеты с большой точностью. У него возникла идея создания универсального вычислителя. В наши дни такие устройства называются компьютерами. Машина Бэббиджа не использовала электричества – работала на шестернях и была механическим устройством.

С развитием электротехники от механических логических элементов перешли сначала к электромеханическим логическим элементам, работающим на электромагнитных реле и затем к электронным логическим элементам на электронных лампах, позже — на транзисторах, к середине XX века.

Современные вычислители построены на транзисторах. В транзисторах, в специальных ячейках памяти сверхбыстрого доступа – регистрах. При этом нам становится важно записать как можно большее число с использованием как можно более простых регистров. Из основных законов информатики можно вывести формулу о зависимости плотности записи информации от основания системы счисления:

$$y = \frac{\ln x}{x}$$

Рисунок 1 Формула плотности информации

В формуле x – основание системы счисления. Функция имеет максимум при значении x равном числу e. Так как люди используют целочисленные системы, то максимальная экономичность показательных позиционных систем счисления достигается при основаниях 2 и 3.

Как писал отечественный математик Н. Фомин: «Экономичность системы счисления — немаловажное обстоятельство с точки зрения её использования в

вычислительной машине. Поэтому, хотя применение в вычислительной машине троичной системы вместо двоичной влечёт некоторые конструктивные трудности (при этом нужно пользоваться элементами, каждый из которых может находиться не в двух, а в трёх устойчивых состояниях), эта система уже была использована в некоторых реально существующих вычислительных устройствах».[1]

Двоичность и троичность позволяет значительно сократить количество операций и элементов, выполняющих эту обработку, по сравнению с десятичными логическими элементами. Начался переход к преимущественно двоичной и троичной системам счисления от десятичных логических элементов.

Двоичная логика является подмножеством троичной логики. В троичной логике помимо ИСТИНЫ и ЛОЖЬ, появляется третье: НЕИЗВЕСТНОСТЬ. Троичная логика позволяет производить математические операции быстрее и делать более короткие записи чисел. Производство компонентов на троичной логике оказалось более сложным технически и обходилось дороже. Поэтому, в настоящий момент троичные компьютеры полностью вытеснены двоичными. Современные формальная логика и техника основаны на существовании только двух состояний.

При построении вычислительной техники используются разные логические устройства, состоящие из элементов, производящих операции над числами или переменными. Логические элементы, можно разделить на два класса:

- Комбинационные (конечные автоматы без собственной памяти)
- Последовательные (дискретные автоматы с памятью)

Логика работы первого типа устройств называется комбинационной, а второго – секвенциальной. В устройства с комбинационной логикой значения выходных сигналов в каждый момент времени полностью определяются значениями сигналов на её входах. Одной и той же комбинации входящих

значений соответствует одна и та же комбинация исходящих. Это возможно из-за полного отсутствия памяти у данного типа устройств. Предыдущее состояние не сохраняются и сигналы, поступившие ранее, не учитываются.

Примерами устройств с комбинационной логикой могут служить:

- Электронные ключи
- Шифраторы и дешифраторы
- Арифметические устройства, в том числе арифметические блоки процессора

В принципе работы логических элементов заложен аппарат математической логики. Все логические элементы, существующие в цифровой электронике, представляет собою реализацию той или иной булевой функции.

При конструировании устройства, мы математически описываем каждый его узел. В модели отсутствует поправка на время взаимодействия элементов схемы. Это одно из упрощений математической модели в отношении реальных схем. В техническом отношении передача и преобразование информации реализуется с помощью булевой двоичной математики: различается только два уровня сигнала, соответствующие понятиям ИСТИНА и ЛОЖЬ, что соответствует наличию и отсутствию тока в цепи. Устройства в цепи обозначаются соответствующими логическими функциями, аргументы и результат функции принимают так же только два значения: 0 или 1. Итог работы схемы это результат функции, так же состоит из двоичного алфавита.

Логическая функция может быть задана разными способами:

- словесно
- таблицей истинности
- аналитически
- графически

Пример словесного описания: функция f имеет два аргумента и принимает значение 1, когда значения аргументов равны. При неравенстве аргументов, функция принимает значение 0. Данный способ задания функции

не требует понимания языка формальной математики, но и делает сложной работу с функцией в данной записи. [6]

Аналитическое описание функции в виде алгебраического выражения. Одну логическую функцию можно записать несколькими вариантами. К преимуществам данного способа записи можно отнести возможность определения структуры логического устройства.

$$f(a,b) = a \cdot \bar{b} \vee a \cdot b$$

Рисунок 2 Аналитический способ записи

Последний способ называется таблица истинности - указывает значения выходных сигналов при всех значениях входящих сигналов. В таблице истинности значения каждого из сигналов располагаются в столбцах, а строки обозначают соответствия между наборами сигналов. Всего 2^n строк, где n – число входов. Таблица истинности позволяет просто и наглядно отразить функциональную зависимость параметров на выходе от параметров на входе. Но по одной таблице невозможно определить структуру логического устройства, смотря на перечень преобразований.

№ набора	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Рисунок 3 Таблица истинности

Для определения структуры логического устройства решают задачу синтеза - разработка логической схемы по её аналитическому описанию.

Противоположна задаче синтеза задача анализа – нахождение функции, реализуемой заданной схемой. Решив задачу анализа комбинационной схемы, мы узнаем и сможем сопоставить входные значения выходным, составить таблицу истинности.[11]

1.2 Начертание основных логических элементов

Логический элемент может иметь любое конечное число входов. Обычно используется не более четырех. У самых распространенных элементов входов два. Выход всегда один, так как логический элемент проводит логическую операцию с единственным ответом. Всего возможно 16 двоичных двухвходовых логических функций. Не все логические функции имеют свой элемент. Опишем 9 основных логических элементов: «отрицание», «дизъюнкция», конъюнкция, «сложение по модулю 2», «эквивалентность», «импликация», «компликация», «И-НЕ», «ИЛИ-НЕ», «исключающее ИЛИ».

В соответствии с требованиями ГОСТ 2.701-84 и ГОСТ 2.702-7, устанавливаются стандарты условно графических обозначений (УГО) для логических функций. В основе обозначения элемента цифровой техники лежит прямоугольник. УГО может содержать основное поле и одно или два дополнительных, расположенных по обе стороны от основного. Размеры УГО зависят «по ширине – от числа полей и меток, расположенных в этих полях» и «по высоте – от числа выводов, интервалов между ними и числа строк в основном и дополнительных полях».[10]

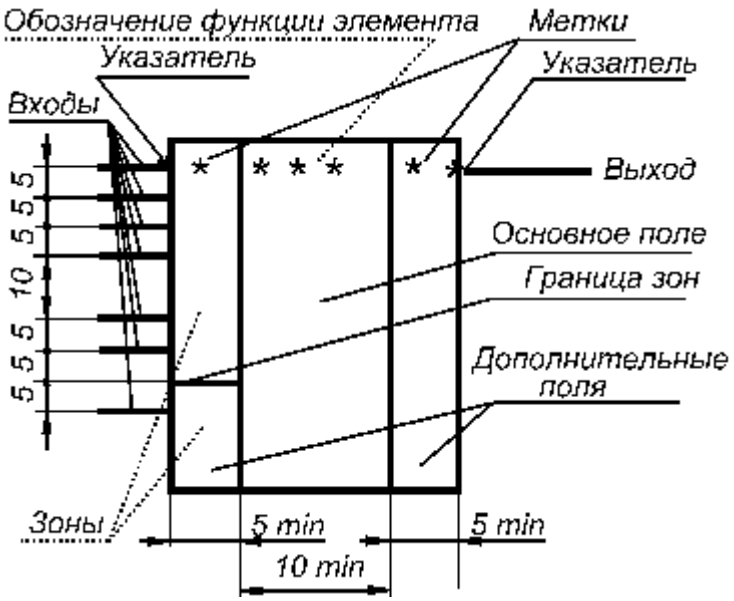


Рисунок 4 Стандарт построения обозначения

Ширина основного поля должна быть не менее 10 мм, дополнительных – не менее 5 мм, расстояние между выводами – не менее 5 мм, расстояние между выводом и горизонтальной стороной (или границей зоны) УГО – не менее 2,5 мм или кратным ему. Допускается увеличивать ширину полей при нанесении большого числа меток и функций. Входы элемента изображаются слева УГО, а выходы – справа. Допускается смена ориентации изображения и поворот на 90 градусов. Выводы можно объединять в группы, которые разделяются интервалом не менее 10 мм или кратным 5 мм.[10]

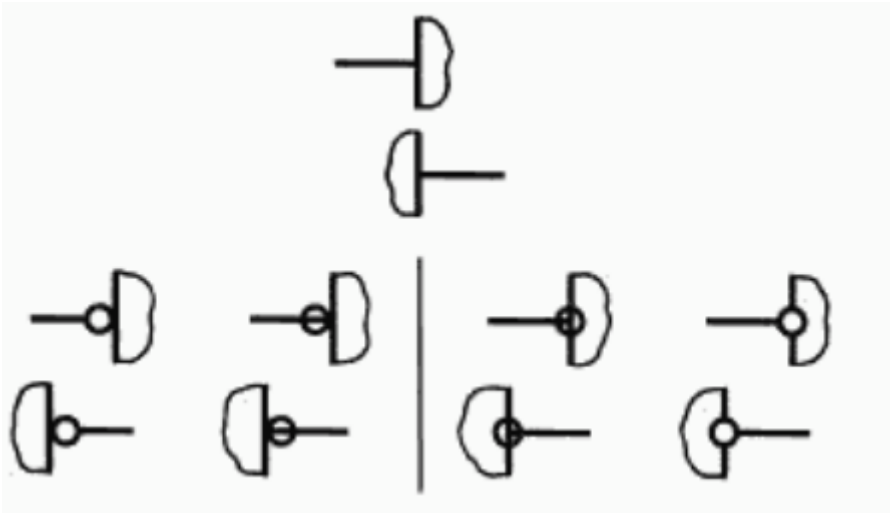


Рисунок 5 Функциональное обозначение выводов

Функциональное обозначение выводов изображается с помощью меток. Так на Рисунок 5 можно увидеть обозначения:

- Прямой статический вход;
- Прямой статический выход;
- Инверсный статический вход;
- Инверсный статический вход.

Кроме отечественного стандарта, существует распространенный в Европе стандарт DIN и американский ANSI. В основе иностранных обозначений лежит треугольник, видоизменяемый, в зависимости от функции.

Начнем с логической функции «**НЕ**» - «*отрицания*». Имеет по одному входу и выходу. Выводит значение, противоположное полученному на входе.

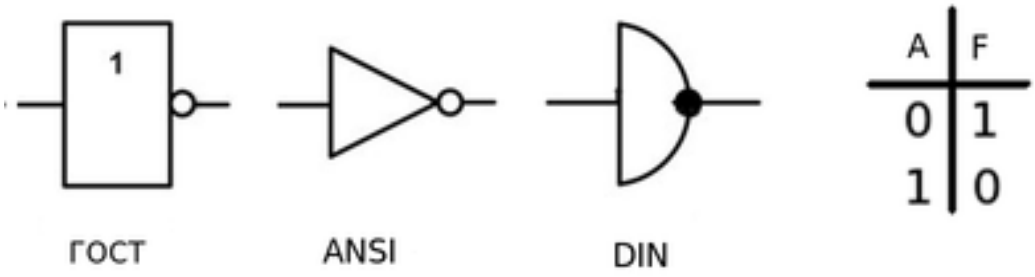


Рисунок 6 Логический элемент «НЕ»

Логический элемент «**И**». Аналогична операции «*конъюнкция*» в математической логике – «логическое умножение». Сигнал на выходе соответствует единице только в случае, когда на все входы поданы сигналы логической единицы.

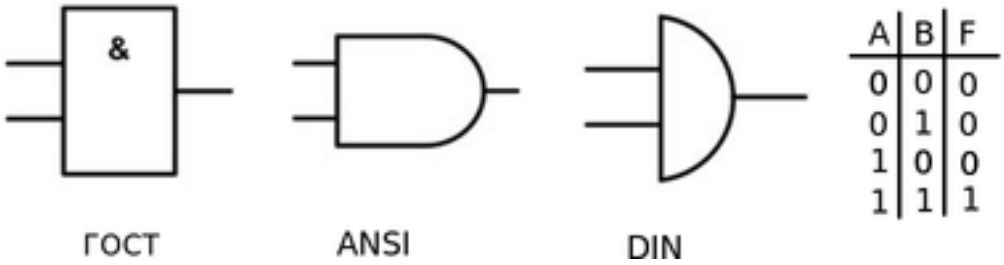


Рисунок 7 Логический элемент «И»

Логический элемент «**ИЛИ**». Эквивалент операции «*дизъюнкция*». Сигнал на выходе равен нолю только в случае, когда на все входы поданы ноли. Элемент обеспечивает логическое сложение входных сигналов.

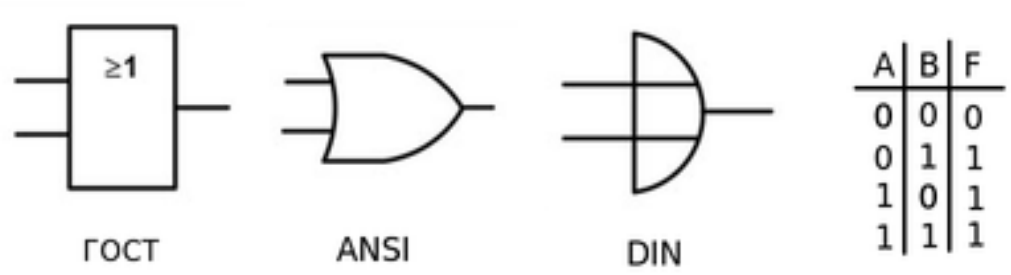


Рисунок 8 Логический элемент «ИЛИ»

Логический элемент «**И-НЕ**». Так же называемый «штрих Шеффера» Элемент реализует инверсию логического произведение всех входных сигналов. Выходной сигнал соответствует нулю только в случае, когда сигналы на всех входах равны единице.

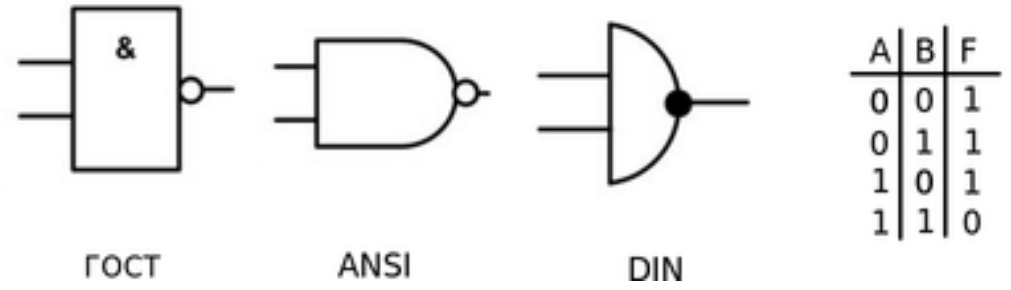


Рисунок 9 Логический элемент «И-НЕ»

Логический элемент «**ИЛИ-НЕ**». В иностранной литературе называется «стрелка Пирса» или «NOR». Единица на выходе получается только если на всех входах ноль. В остальных случаях на выходе ноль.

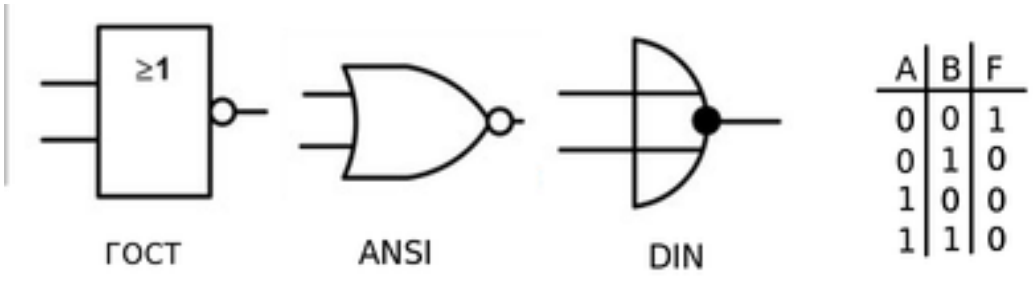
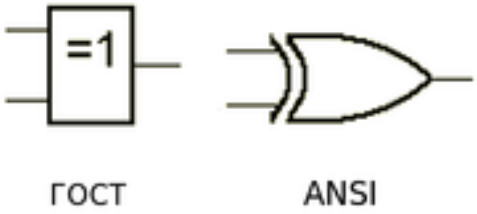


Рисунок 10 Логический элемент «ИЛИ-НЕ»

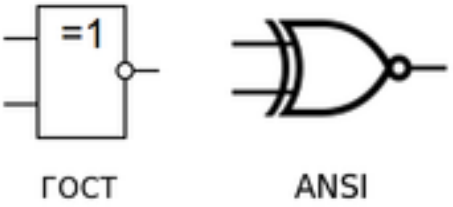
Логический элемент «**Исключающее ИЛИ**», широко применяемый в криптографических алгоритмах, известный как XOR. Подсчитывается число, получаемое на входе. При четном числе входящих единиц, на выходе ноль. При нечетном – единица.



A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

Рисунок 11 Логический элемент «Исключающее ИЛИ»

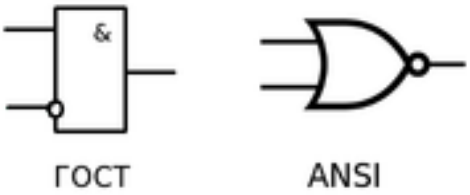
Эквивалентность. На выходе получается единица, если на всех входах одинаковые значения.



A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

Рисунок 12 Эквивалентность

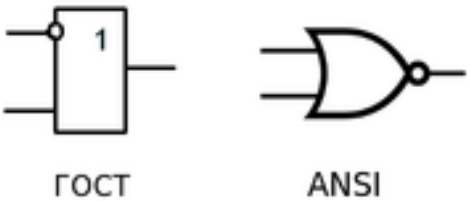
Прямая импликация. От A к B. Ноль только когда на выходе B меньше, чем на A.



A	B	F
0	0	1
0	1	1
1	0	0
1	1	1

Рисунок 13 Импликация

Обратная импликация. От B к A. Ноль только когда на выходе B больше, чем на A.



A	B	F
0	0	1
0	1	0
1	0	1
1	1	1

Рисунок 14 Обратная импликация

Базисом являются функции отрицание, дизъюнкция, конъюнкция. С их помощью мы можем выразить другие функции, представленные выше. Например, импликация можно выразить через отрицание и дизъюнкцию:

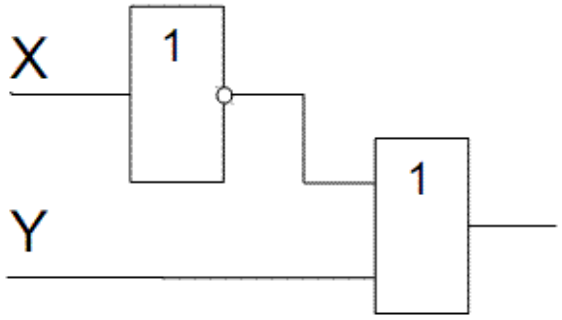


Рисунок 15 Импликация через отрицание и дизъюнкцию

1.3 Существующие программы для построения комбинационных схем

В сети Интернет можно найти несколько разных программ по данной теме. Многие из них созданы для обучения самой математической логике и булевой алгебре и непригодны для полноценного конструирования схем.

Тренажер «Логика»

Официальный сайт http://kpolyakov.spb.ru/prog/logic.htm

Данная программа создана для проведения практических занятий по теме «Математическая логика» в игровой форме и ориентирована на учебную программу 9-го класса общеобразовательных школ. Программа работает под управлением операционных систем линейки Windows 95/98/NT/2000/XP/2003. Обновления давно не выходили. Программа бесплатна для некоммерческого использования. В программе реализован базовый набор логических элементов: И, ИЛИ, НЕ, в схемах можно использовать включенные элементы импликация, эквивалентность, а также полусумматор, сумматор и RS-триггер. Исходные код программы не распространяется.

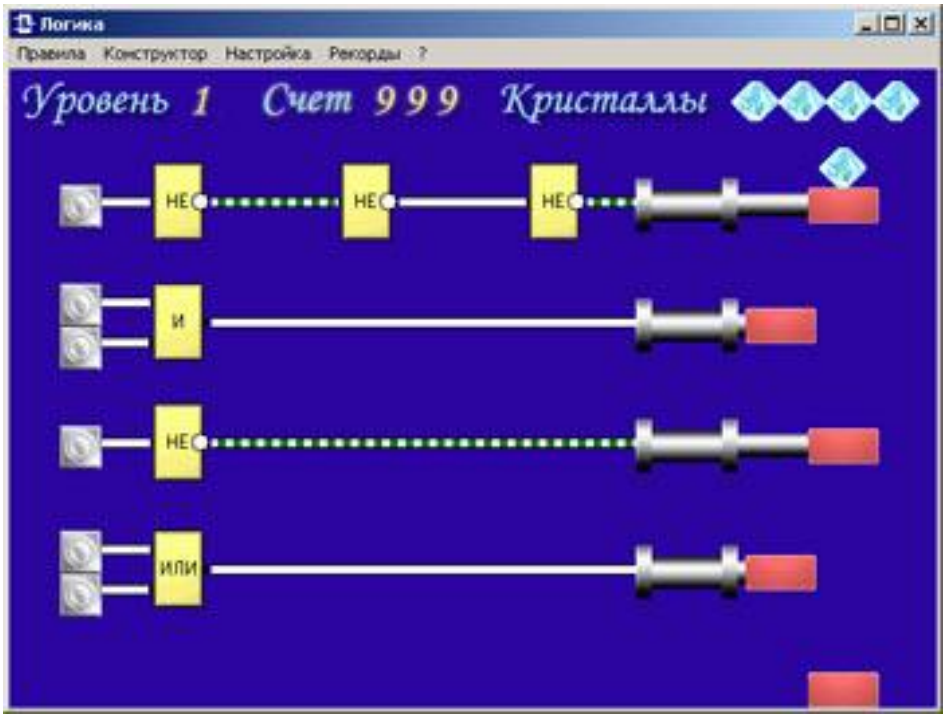


Рисунок 16 Тренажер "Логика"

Имеется встроенный набор задач - логических схем. Существует возможность составлять простые схемы с помощью конструктора и проверять их работу. В настройках можно выбрать уровень для загрузки. Конструктор схем сделан в виде текстового редактора. Необходимо знать как пишется схема, в нужном для программы, формате.

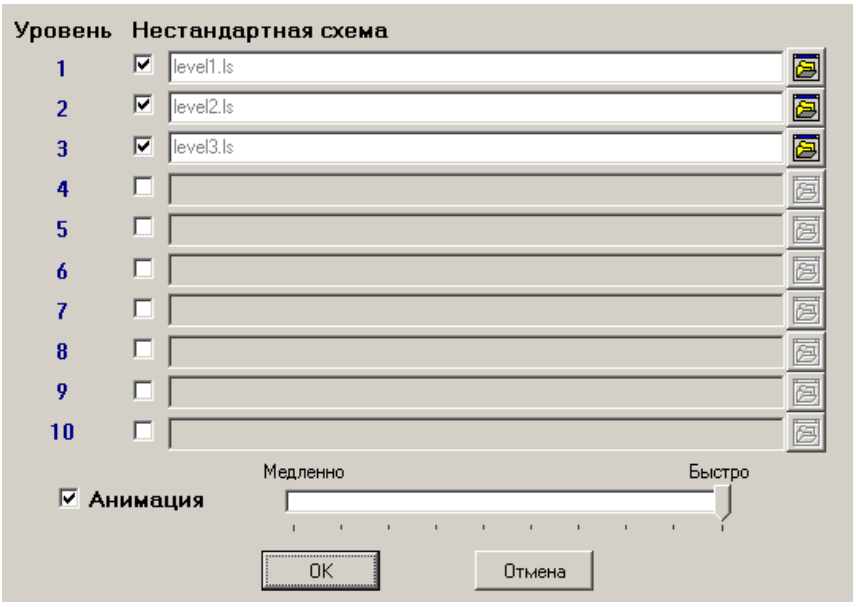


Рисунок 17 Настройки тренажера "Логика"

Онлайн конструктор Logic

Официальный сайт http://logic.ly/demo/

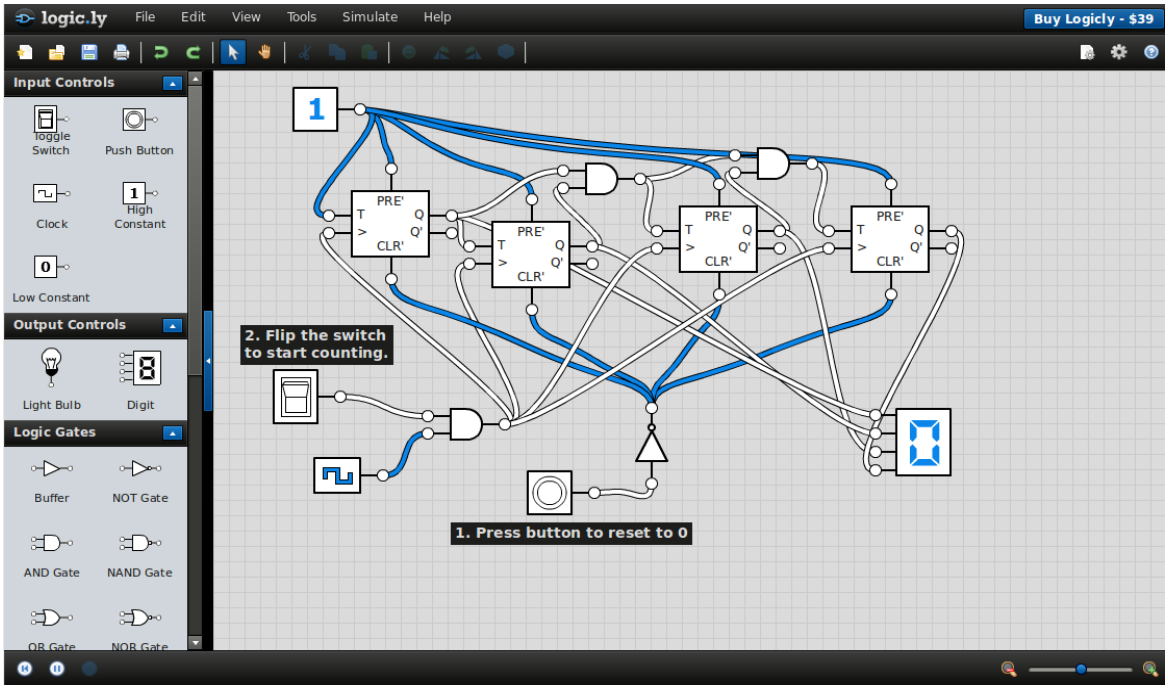


Рисунок 18 Рабочее пространство Logic.ly

Программа работает в браузере и требует установленный Adobe Flash Player. Данный конструктор предназначен для проведения занятий в учебных заведениях. Интерфейс выглядит современным и понятным. Интерфейс на английском. При запуске программы, мы видим рабочее пространство и набор логических элементов слева. Элементы оформлены по стандарту ANSI, так как разработчик программы из США. Предоставлен более широкий выбор элементов, чем в предыдущей программе. Можно использовать базовый набор логических элементов, а так же несколько логических элементов с памятью, световое табло для отображения чисел. Конструктор позволяет построить любые схемы из представленных элементов. Программа выглядит серьезнее и предназначена для пользователей старшего возраста.

Программа платная. Полная версия стоит 39$

Electronic Workbench

Официальный сайт http://www.ni.com/

Программа работает на компьютерах под управлением ОС Windows. Electronics Workbench - профессиональный инструмент, позволяющий

моделировать аналоговые, цифровые и цифро-аналоговые схемы большой степени сложности. Программа поставляется с библиотеками, содержащими все широко распространенные компоненты. Есть возможность подключения и создания новых библиотек компонентов. Параметры компонентов можно изменять в широком диапазоне значений.

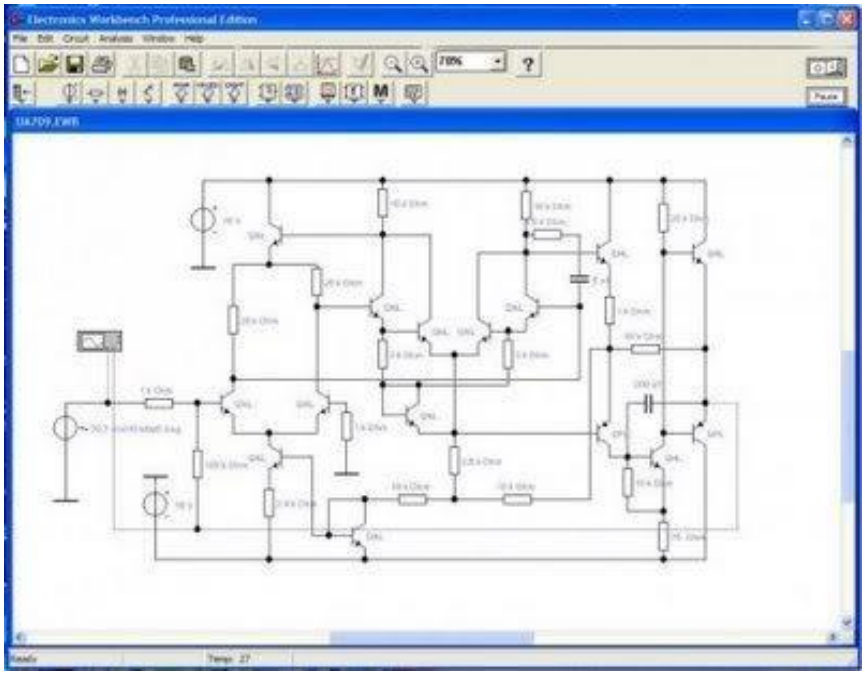


Рисунок 19 Электронная схема в Workbench

Программа способна заменить собой лабораторию и сделать изучение электрических схем доступным в домашних условиях. Каждый компонент описываются набором параметров, значения которых можно задавать непосредственно с клавиатуры, активные элементы - моделью, представляющей собой совокупность параметров и описывающей конкретный элемент или его идеальное представление. Модель выбирается из списка библиотек компонентов, параметры модели также могут быть изменены пользователем.

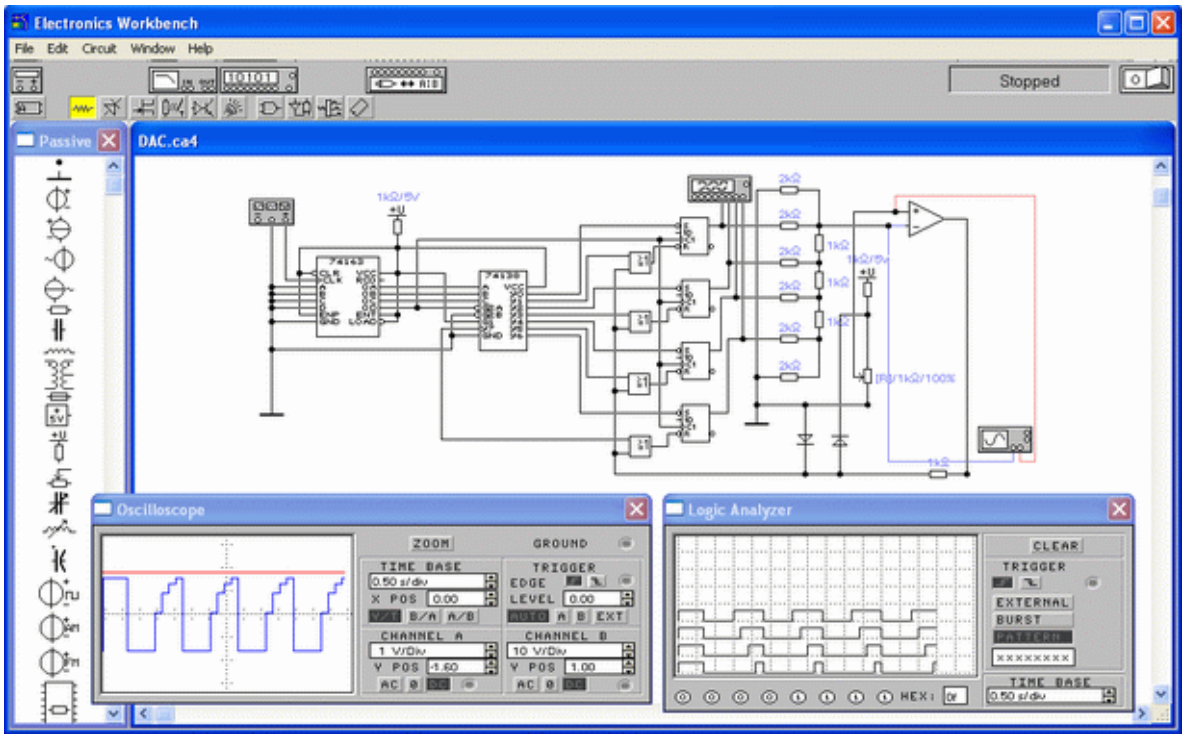


Рисунок 20 Режим эмуляции действующей схемы с графиками

Широкий набор приборов позволяет производить измерения различных величин, задавать входные воздействия, строить графики. Все приборы изображаются в виде, максимально приближенном к реальному. Результаты моделирования можно вывести на принтер или импортировать в текстовый или графический редактор для их дальнейшей обработки. Рабочее пространство программы позволяет:

- выбирать элементы и приборы из библиотек,
- перемещать элементы и схемы в любое место рабочего поля, присваивать элементам условные обозначения, поворачивать элементы и их группы на углы, кратные 90 градусам, копировать, вставлять или удалять элементы, фрагменты схем,
- изменять цвета проводников, выделять цветом контура схем,
- одновременно подключать несколько измерительных приборов и наблюдать их показания на экране монитора, задавать режим работы прибора,

Программа платная, стоимость лицензии начинается от 700$.

Две программы написаны для работы под ОС Windows. Это делает сложной запуск данных программ под ОС на базе Linux.В настоящий момент, Linux активно продвигается на государственном уровне. Согласно авторитетному новостному сайту РБЦ «С 1 января 2016-го российские госорганы могут закупать импортный софт только в тех случаях, если у него нет отечественного аналога»[9]. Многие крупные российские разработки ОС строятся на базе Linux. Например, Astra Linux, Заря. Astra Linux создана для нужд спецслужб и силовых ведомств. Заря — семейство ОС для использования в Вооружённых Силах РФ. Разработана в Объединённой приборостроительной корпорации по заказу Министерства обороны РФ. Для бюджетного сектора создан Alt Linux. Так же, Alt Linux состоит в списке отечественного ПО и рекомендован для госзакупок. Возможность беспроблемного запуска программ под Linux будет востребована в ближайшее время.

В рассмотренных программах не хватает оформления схем согласно ГОСТ. Тренажеры для учащихся не задуманы для серьезной работы: элементы изображаются искаженно и упрощенно. Зарубежные программы используют другие стандарты для обозначения логических элементов. Добавление обозначений в отечественном формате сделает возможным использование программы в работе на предприятиях.

Среди рассмотренных программ, ни у одной нет в свободном доступе исходного кода программы. Нет возможности что-то изменить или доработать в самой программе. Если потребуется поменять часть функционала, то придется разрабатывать программу с чистого листа или договариваться с разработчиком, что не всегда удается. Соответственно, необходимо создать программу с открытым исходным кодом. Такой подход позволит заинтересованным в программе людям подстраивать программу под свои задачи и требования. Вырастет интерес со стороны людей, которые будут использовать программу. Мы получим обратную связь, возможно, получится выстроить сообщество

вокруг проекта. Так же исходный код может существенно дополнять документацию и сам служить документацией.

Из известных проектов с открытым исходным кодом можно назвать ядро Linux и браузер Mozilla Firefox. Первое что мы видим на официальном сайте Linux – исходный код. Этим проектам удалось стать известными благодаря такому подходу.

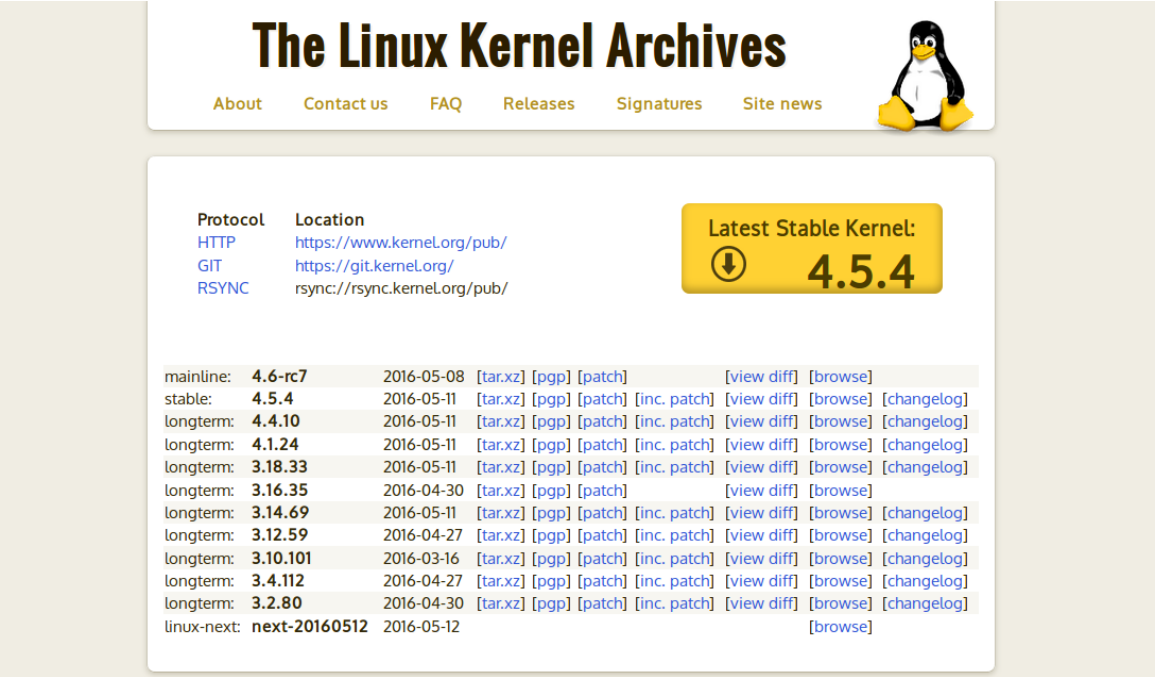


Рисунок 21 Главная страница официального сайте Linux

Еще одна полезная функция: наличие пакетного режима работы. Возможность обработки или взаимодействия с программой без графического интерфейса. Пакетный режим полезен при автоматизации каких-то рутинных действий. Необходимо обеспечить часть функционала через параметры при запуске. В дальнейшем, при необходимости, добавить поддержку консольного интерфейса.

Комбинационные схемы построены на базе алгебры логики, которая является основным математическим аппаратом и легко реализуема с помощью информационных технологий. Это позволяет полностью автоматизировать обработку уже готовой схемы. Для пользователя останется сделать удобный интерфейс, проверяющий корректность ввода и не допускающий очевидные

ошибки. В конструкторе схем можно автоматизировать задачу анализа: построение таблицы истинности для заданной пользователем схемы. Добавить проверку корректности.

Существует большое число программ для обработки комбинационных схем, но нет продукта, который бы подходил для проведения лабораторных работ в высших учебных заведениях, был бы гибок для разных учебных программ. Необходим удобный инструмент, который можно легко изменить под учебную программу и применять.

2. Разработка программы

2.1 Формализованное описание технического задания

1. Общие сведения

1.1. Название организации-заказчика

Проектирование и разработка программно-технологического комплекса для обучения в высшей школе осуществляется в рамках научно-исследовательской деятельности ИМИиИТ.

1.2. Название продукта разработки (проектирования)

Программно-технологический комплекс для построения комбинационных схем.

1.3. Назначение продукта

Комплекс предназначен для расширения возможности применения вычислительных устройств (стационарные компьютеры и ноутбуки) при проведении учебного процесса, связанного с освоением студентами вуза теоретических разделов учебных дисциплин: информатики и математической логики.

1.4. Плановые сроки начала и окончания работ

Начало работ: 01 сентября 2015 г.; окончание работ 18 мая 2016 г.

2. Характеристика области применения продукта

2.1. Процессы и структуры, в которых предполагается использование продукта разработки:

Программно-технологический комплекс предполагает возможность использования в учебном процессе высших учебных заведений, колледжей.

2.2. Характеристика персонала (количество, квалификация, степень готовности)

Продукт ориентирован на использование профессорско-преподавательским составом вуза и студентами.

3. Требования к продукту разработки

3.1. Требования к продукту в целом

3.1.1. Структура программного обеспечения

Предлагается выделить следующие функциональные подсистемы и компоненты:

- Режим пакетной обработки;
- Графический интерфейс пользователя;
- Инструкция для пользователей по использованию программного обеспечения.

3.1.2. Программное обеспечение должно обеспечивать:

- Работу под операционными системами на базе Linux;
- Возможность простого расширения функционала и добавления новых элементов;
- Сохранение результатов работы пользователя.

3.2. Аппаратные требования

Для программного обеспечения требуются аппаратные средства, соответствующие следующим минимальным системным требованиям требования

- Процессор с тактовой частотой 1500 MHz.
- Оперативная память 512 Мб.
- Свободное место на жёстком диске в размере 500 Мб.
- Наличие графического ускорителя.
- Возможность работы приложения без доступа к сети Интернет.

4. Требования к документированию

4.1. Перечень сопроводительной документации

Сопроводительная документация включает инструкцию по установке, руководство пользователя для освоения работы в данном программном обеспечении.

2.2 Выбор технологий

При выборе инструментов нужно учесть множество факторов. Начнем с самого строгого критерия. В данном случае это необходимость обеспечить работу программы под разными платформами: ОС на базе Linux и Windows.

В настоящее время часто этот вопрос решается с помощью создания сайта. Браузеры есть под каждой платформой. Даже на телефонах встречается полноценный браузер. Стандарты и технологии позволяют сделать сайт одинаково работающим на любой ОС. Разработка приложения, в таком случае разделяется на две больших части: Front-end и Back-end. Первая часть состоит в представлении данных, описании интерфейса пользователя. Код выполняется на компьютере пользователя. Вторая часть занимается обработкой и хранением данных. Код выполняется на сервере. Популярные языки программирования Front-end это JavaScript, Back-end – PHP.

Такое разделение усложняет разработку, так как нам требуется несколько специалистов разных направлений, требуется настройка и поддержание работы сервера. Кроме того, из браузера мы имеем очень ограниченный доступ к системе: нам перекрыт доступ к файловой системе, затруднен доступ к ресурсам устройства. Требуется самая новая версия браузера. Браузер добавляет один уровень абстракции от компьютерного железа, что снижает производительность программы. Приложения получаются менее отзывчивые и не такие функциональные, как обычные – нативные - приложения для компьютера, скомпилированные именно под эту платформу. К тому же, для сайта требуется настроенная компьютерная сеть. Соответственно, формат сайта не подходит для нашего приложения.

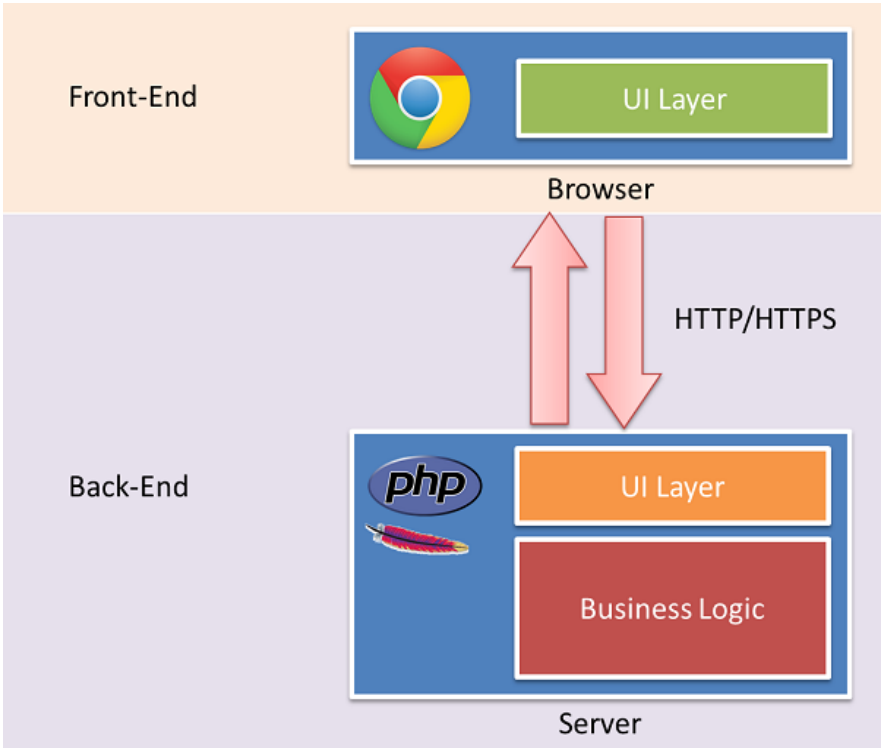


Рисунок 22 Front-End и Back-End

Для нативного приложения нужно выбрать язык программирования и библиотеку для отображения графического интерфейса. На разных ОС стандартные элементы выглядят по-разному и при этом важно соблюсти единый внешний вид. Это позволит пользователю не переучиваться и упростит техническое сопровождение программы.

Популярный язык для разработки кроссплатформенных приложений Java. Для Java есть широкий выбор интерфейсных библиотек. Например, Swing. К плюсам можно отнести возможность при разработке расставлять элементы мышкой, как это делается в Delphi. К минусам: приложение будет выглядеть чужеродно в системе, не будет использовать системное оформление. Так же, для самой Java требуется установленная виртуальная Java-машина - Java Runtime Environment. Компания Oracle, обладатель прав на Java, отозвала лицензию на право распространения с дистрибутивами Linux Java-машины в 2011 году[29]. Для работы java-приложений необходимо пользоваться свободным вариантом виртуальной машины OpenJDK или самостоятельно загружать с официального сайта. На платформе Linux не принято загружать

программное обеспечение со сторонних сайтов. Возникают лишние сложности для пользователя при установке и настройке.

Другой популярный язык программирования это Python. Это высокоуровневый интерпретируемый язык. Интерпретатор Python работает под ОС Windows, Linux и MacOS. Высокая скорость разработки и легкое сопровождение получившейся программы — еще одна сильная сторона языка. Синтаксис языка — простой, понятный и лаконичный.

Python активно используется в академической и научной деятельности. Для этого языка написаны несколько научных библиотек для обработки данных, такие как NumPy - альтернатива MATLAB, SciPy – набор функций для выполнения научных и инженерных расчётов, Sci-kit - реализует алгоритмы машинного обучения. Python часто используется для разработки плагинов в различных графических и 3D-системах, мессенджерах, текстовых процессорах, издательских системах и бизнес-приложениях. Python входит в состав большинства дистрибутивов Linux по-умолчанию.

Для Python есть несколько инструментов для создания графического интерфейса: Tk, Qt, wxWidgets.

Tk используется для быстрого создания графического интерфейса. Например, для скриптов с несколькими элементами ввода. В Tk появилась поддержка тем оформления совсем недавно. До этого программы на Tk имели оформление в классическом стиле Windows, что резко выделяло их на фоне большинства современных программ. Tk хорош своей простотой и он недостаточно гибок для написания удобного интерфейса.

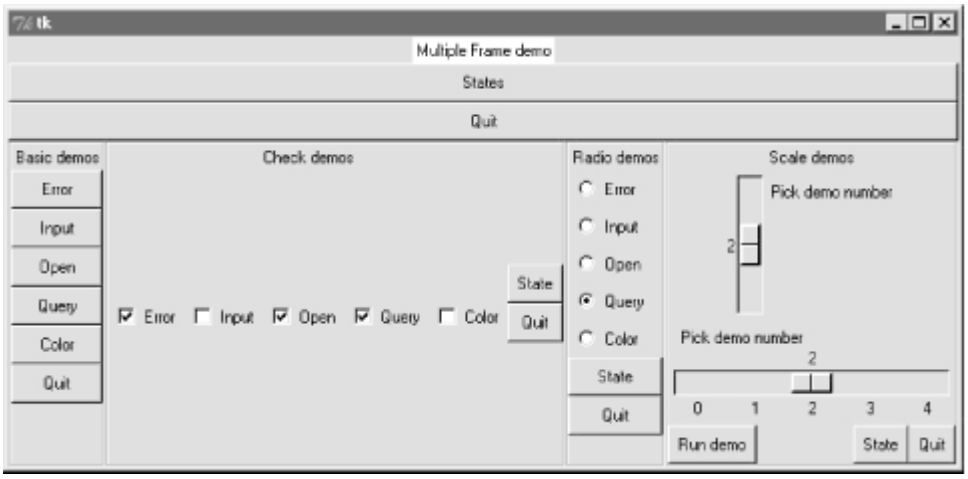


Рисунок 23 Пример интерфейса с применением Tk

Выбор между wxWidgets и Qt сложнее. Так как это две очень популярные библиотеки. Например, с использованием wxWidgets написаны Audacity FileZilla. На Qt написан графическая оболочка VirtualBox, Opera, окружение рабочего стола KDE. Функционал и возможности Qt больше. Qt включает в себя все основные классы, которые могут потребоваться при разработке прикладного программного обеспечения: начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML. Qt является полностью объектно-ориентированным, легко расширяемым. Поэтому выбор сделан в его сторону. Для графического интерфейса будет использована библиотека Qt, точнее её реализация для Python PyQt, представляющая собой набор оберток над стандартными классами графической библиотеки Qt для языка Python

2.3 Проектирование

Составим функциональную модель будущей программы. У пользователя будет цель: составить схему или получить таблицу истинности для заданной схемы. Пользователь собирает схему. Программа выводит таблицу истинности. Контроль будет производиться, по возможности, преподавателем и самой программой. Руководствоваться при составлении схемы правилами двоичной логики и ГОСТ, по части оформления.

Рисунок 24 Диаграмма IDEF0

Программа будет базироваться на стандартах описываемой области. При этом будут использоваться следующие стандарты:

- ГОСТ 2.743-91;
- ГОСТ 2.708-81.

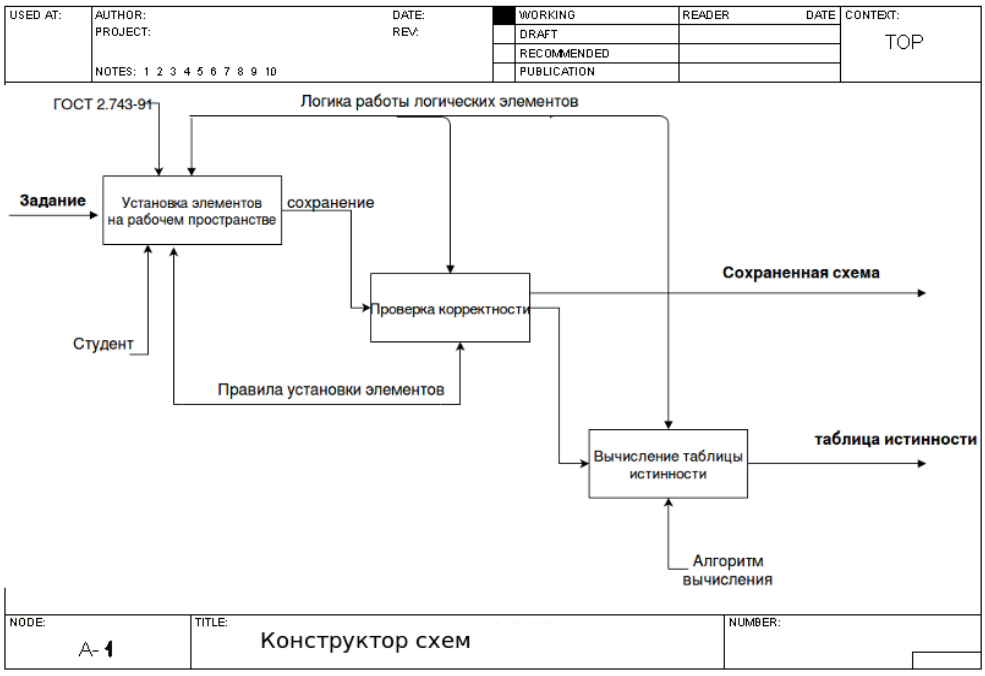


Рисунок 25 Декомпозиция IDEF0

Пользователь будет составлять в рабочей области программы схему из доступных логических элементов. Важно сделать интерфейс так, чтобы он помогал пользователю размещать элементы правильно и указывал на ошибки. Выходные и входные элементы необходимо называть или давать индексы. Так же, обязательно подсвечивать контакты, доступные для соединения. После завершения расстановки элементов, схему нужно сохранить. Если пользователь хочет увидеть результат – таблицу истинности, программа осуществляет проверку корректности схемы:

При положительном завершении проверки корректности, запускается подсчет схемы, составляется таблица истинности. Схему удобно представить внутри программы как ориентированный связный граф и будем проверять:

- логические элементы подключены правильно, отсутствие соединений между двумя выходами или двумя входами одного логического элемента, проверка направленности графа:
- отсутствие петель в графе;
- отсутствие неподключенных контактов или элементов - граф связный.

После проверки, сохраняем граф в виде списка ребёр. Получается файл, каждая строчка которого содержит номера двух смежных вершин. Начинаем обходить граф от входных элементов, к выходным. Получается цикл, в котором мы проходим от по порядку, проверяя выходные значения на каких элементах мы можем вычислить в данный момент. После вычисления значения на выходном соединении, мы удаляем данный элемент из очереди, так как он уже участвовал в вычислении. Когда очередь становится пустой, вычисление считается законченным, выводится таблица истинности.

Программа будет состоять из четырех основных модулей:

- подсчет таблицы истинности;
- графический интерфейса;
- логика элементов;

- проверка корректности схемы.

Для описания логических элементов будет использована парадигма объектно-ориентированного программирования. Это позволит легко добавлять новые логические элементы с помощью наследования. Определим базовый класс Logic, с основными полями и методами. По сути, этот класс представляет собой интерфейс для работы с элементами. Все остальные логические элементы будут наследоваться от него.

Класс Logic содержит поля:

- count_input - количество входов для элемента
- count_output - количество выходов для элемента
- image_file - имя файла с картинкой для элемента из каталога img
- coord_form – если картинка отсутствует, то можно изобразить элемент в виде полигона, указывая список точек в двумерных координатах от 0 до 1 включительно.
- coord_conn – координаты точек соединений, указанные в формате двумерных координатах от 0 до 1 включительно, с третьим параметром
- name – внутреннее название элемента
- writed_name – надпись, которая пишется рядом с элементом на схеме.

И методы класса Logic:

- setValues – устанавливает соответствие между значениями и соединениями при подсчете таблицы истинности:
- calc – возвращает значение логической функции на выходе. Этот метод необходимо переопределять при добавлении своего элемента.

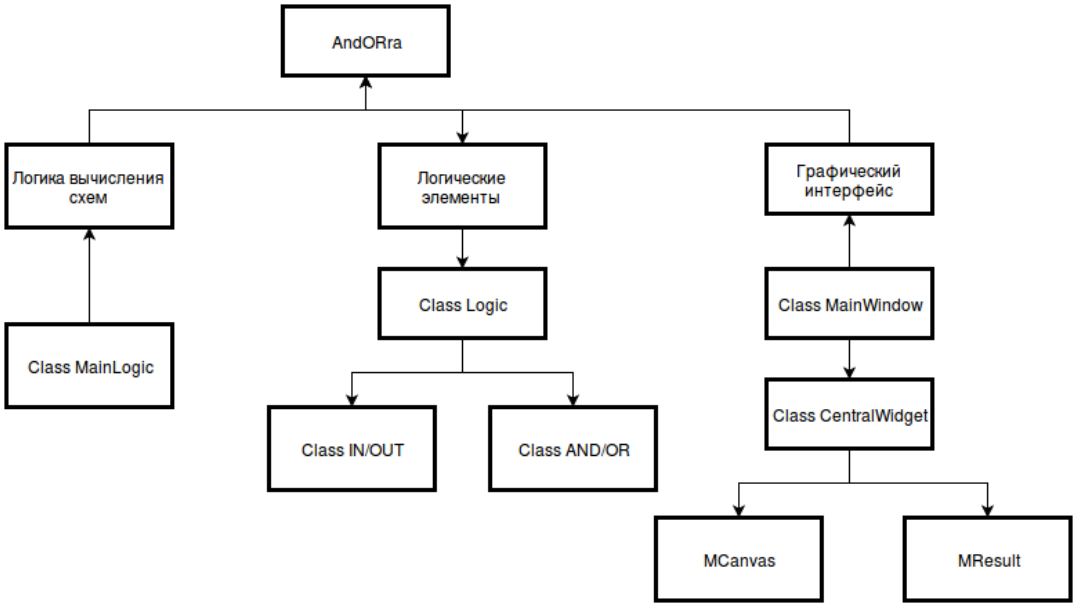


Рисунок 26 Структура классов программы

Логика составления графа и построения таблицы истинности вынесена в класс MainLogic. В классе три функции: fileParser – разбор входного файла при работе в пакетном режиме, genInputValues – генератор разных вариантов входных сигналов, calcScheme – непосредственно функция вычисления результата одной строчки в таблице истинности по заданному входному набору значений.

Создает окно программы класс MainWindow. В этом классе определен только конструктор, функция создания меню и строки состояния программы. Далее MainWindow создает экземпляр класса CentralWidget, являющегося наследуемым от стандартного в Qt класса QWidget. CentralWidget добавляет остальные элементы интерфейса: кнопки с доступными элементами, рабочее пространство – экземпляр класса MCanvas и место для вывода таблицы истинности – экземпляр класса MResult. Класс CentralWidget содержит набор методов для сохранения результатов работы пользователя в файл, в формате программы

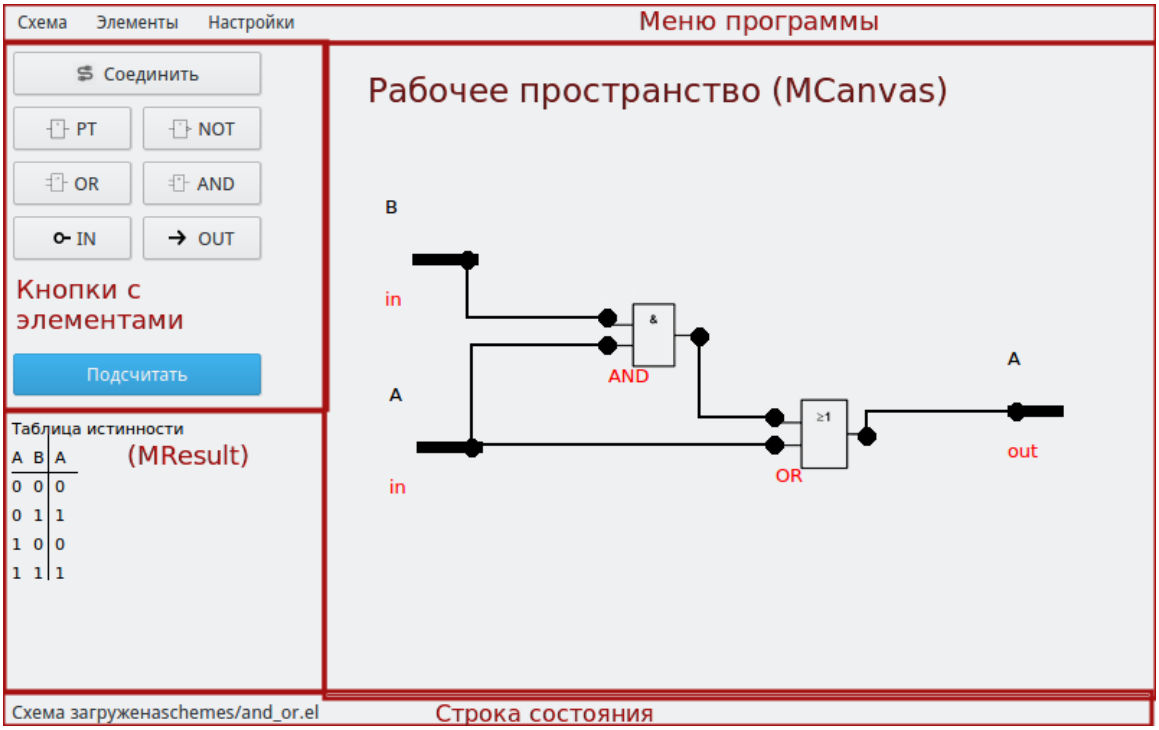


Рисунок 27 Главное окно программы с выделением классов

Класс MResult содержит всего три функции. Функция setValueTable, в которой объект получает значения таблицы истинности, конструктор и функция paintEvent, в которой происходит отображение значений таблицы на экране.

Класс MCanvas самый большой, обрабатывает движения курсора и нажатия кнопок мышки. Так же здесь происходит отрисовка всех элементов, проверка построения схемы. Элементы рисуются в несколько этапов:

- Проверка картинки с отображением элемента и её отображение на экране в заданном месте. Если картинки нет, то отображается полигон, по заданным в элементе точкам;
- Рисуются места соединений, в виде серых кругов;
- Отображается надпись – название элемента. Для элементов входа и выхода также отображаются их названия;
- Отрисовываются соединения между элементами.

При необходимости, можно увеличить размер объектов, правкой поля класса с именем Delta. Это ширина элемента и её значение используется при

вычислении размеров остальных элементов. Вся схема, кроме элементов, это векторная графика. Не используются никакие системные элементы. Это позволяет полностью контролировать внешний вид, масштабировать при необходимости и сделать его одинаковым под любой ОС.

В классе MCanvas происходит проверка корректности ввода. Например, нельзя соединить элемент сам с собой, соединить входы и выходы двух элементов.

Интерфейс выглядит современно и лаконично. Часто используемые функции вынесены на отдельные кнопки. Используются термины и обозначения из предметной области. Интерфейс не реагирует негативным образом на неумелые действия пользователя благодаря проверкам и контролю ввода. Обратная связь с пользователем осуществляется через строку состояния. Периодически, там указывается, что программа ожидает от пользователя.

2.4 Техническая документация к программе для пользователей

1. Что такое «AndORra».

«AndORra» – программное обеспечение, предназначенное для создания графических схем и вычисления по схеме таблицы истинности. Результат работы программы, представляет собой таблицу истинности.

2. Отличительные черты «AndORra»

Интуитивно-понятный графический интерфейс, который быстро и легко осваивается. Используется управление мышью для конструирования схемы. Пользователю не нужно знать что либо, кроме самой предметной области. Сторонние знания не требуются. Наличие подсказок в строке состояния помогает понять, что программа ожидает и какие действия были выполнены.

Программа «AndORra» автоматизирует процесс конструирования комбинационных схем, помогая пользователю следить за правильностью расстановки элементов, прививает знания маркировки элементов по ГОСТ.

3. Минимальные системные требования для использования программы «AndORra».

Операционная система на базе Linux, либо MacOS, либо Windows (XP, 7, 8 и выше) с установленным набором библиотек PyQt и интерпретатором Python версии 3.

4. Установка необходимых библиотек:

Для ОС на базе Linux:

Необходимо установить PyQt и Python в пакетном менеджере. Скорее всего в дистрибутиве есть какой-то графический пакетный менеджер. Но в общем случае установку лучше производить из терминала. Вам потребуются права root для установки. Если у вас есть системный администратор, то лучше попросить его сделать это самостоятельно

a) Запустите терминал, нажав ctrl+alt+T

b) Введите строку:

sudo apt install python3-pyqt5 python3-dev –y

c) Введите пароль администратора.

Все необходимые библиотеки должны установиться автоматически.

```
voidarray ~ # sudo apt install python-pyqt5 python3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-dev is already the newest version.
The following extra packages will be installed:
  libatk1.0-0 libatk1.0-data libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libdrm2 libegl1-mesa
  libegl1-mesa-drivers libelf1 libgbm1 libgl1-mesa-dri libgl1-mesa-glx libglapi-mesa
  libgtk2.0-0 libgtk2.0-bin libgtk2.0-common libice6 libllvm3.5 libmtdev1 libopenvg1-mesa
  libpciaccess0 libqt5clucene5 libqt5core5a libqt5dbus5 libqt5designer5 libqt5gui5 libqt5help5
  libqt5network5 libqt5printsupport5 libqt5sql5 libqt5sql5-sqlite libqt5test5 libqt5widgets5
  libqt5xml5 libsm6 libtxc-dxtn-s2tc0 libwayland-client0 libwayland-egl1-mesa
  libwayland-server0 libx11-xcb1 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-icccm4
  libxcb-image0 libxcb-keysyms1 libxcb-present0 libxcb-randr0 libxcb-render-util0 libxcb-shape0
  libxcb-sync1 libxcb-util0 libxcb-xfixes0 libxcb-xkb1 libxcomposite1 libxcursor1 libxdamage1
  libxfixes3 libxi6 libxinerama1 libxkbcommon-x11-0 libxkbcommon0 libxrandr2 libxshmfence1
  libxxf86vm1 python-sip qttranslations5-l10n x11-common xkb-data
Suggested packages:
  librsvg2-common gvfs pciutils python-pyqt5-dbg
The following NEW packages will be installed:
  libatk1.0-0 libatk1.0-data libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libdrm2 libegl1-mesa
  libegl1-mesa-drivers libelf1 libgbm1 libgl1-mesa-dri libgl1-mesa-glx libglapi-mesa
  libgtk2.0-0 libgtk2.0-bin libgtk2.0-common libice6 libllvm3.5 libmtdev1 libopenvg1-mesa
  libpciaccess0 libqt5clucene5 libqt5core5a libqt5dbus5 libqt5designer5 libqt5gui5 libqt5help5
  libqt5network5 libqt5printsupport5 libqt5sql5 libqt5sql5-sqlite libqt5test5 libqt5widgets5
  libqt5xml5 libsm6 libtxc-dxtn-s2tc0 libwayland-client0 libwayland-egl1-mesa
  libwayland-server0 libx11-xcb1 libxcb-dri2-0 libxcb-dri3-0 libxcb-glx0 libxcb-icccm4
  libxcb-image0 libxcb-keysyms1 libxcb-present0 libxcb-randr0 libxcb-render-util0 libxcb-shape0
  libxcb-sync1 libxcb-util0 libxcb-xfixes0 libxcb-xkb1 libxcomposite1 libxcursor1 libxdamage1
```

Рисунок 28 Установка PyQt и Python в Linux

Для Windows:

Установим Python. Необходимо загрузить подходящую версию интерпретатора на официальном сайте: http://www.python.org/. Версия Python должна быть не ниже 3. Разрядность - соответствующую разрядности вашей системы. Если сомневаетесь в разрядности установленной системы, то эту информацию можно узнать нажав WIN+Pause (WIN это клавиша с флагом на клавиатуре).

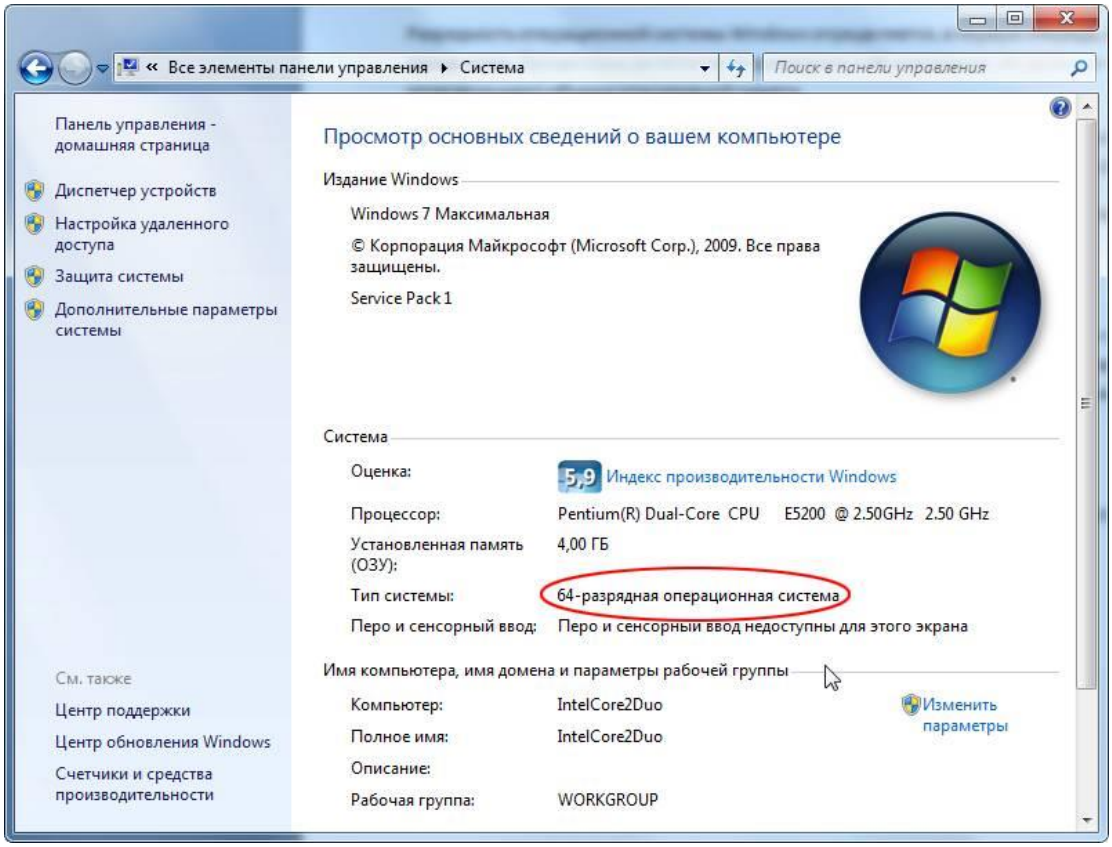


Рисунок 29 Окно с информацией о разрядности системы

Осталось установить PyQt. Его можно загрузить с официального сайта https://www.riverbankcomputing.com/software/pyqt/download5. Выбираете инсталлятор соответствующей разрядности. Разрядность Python и PyQt должны совпадать.

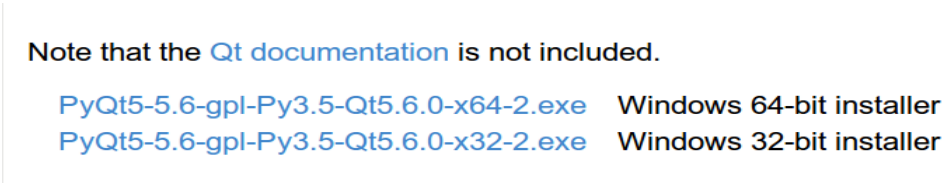


Рисунок 30 Выбор инсталлятора PyQt

После установки необходимых программ, можно запускать «AndORra». Для запуска необходимо служит файл start.py, находящийся в корневом каталоге программы.

5. Основные инструкции по использованию программы «AndORra»:

При запуске программы, открывается главное окно программы. Функционально окно можно разделить на две части. В левой части находятся кнопки управления: вверху список элементов, основные функции, внизу поле для вывода таблицы истинности. Большую часть окна занимает рабочее пространство для размещения логических элементов. Вверху расположено меню со всеми функциями.

Рисунок 31 Главное окно после запуска программы

Создание схемы в AndORra можно разделить на несколько этапов.

Первый этап: расстановка необходимых логических элементов. В данной версии программы их 5: «И», «ИЛИ», «НЕ» и «повторитель» Так же два элемента для обозначения входа и выхода схемы – “IN” и “OUT”. Для размещения элемента на рабочем поле необходимо нажать кнопку нужного нам элемента и затем нажать левой кнопкой мыши на поле, где хотим поместить элемент. После клика левой кнопкой мыши, на рабочем пространстве появится элемент. Места контактов выделены кругами серого цвета. Контакт, рядом с

которым находится курсор, подсвечивается желтым цветом. Это будет особенно полезно на втором этапе, при связывании элементов между собой.

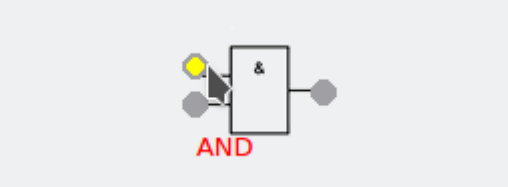


Рисунок 32 Пример элемента на рабочем пространстве

Второй этап: связывание нужных элементов в правильном порядке. Простой расстановки не достаточно, так как надо знать путь, по какому должен идти сигнал. Связывание элементов делается следующим образом:

1. Разместить два элемента:

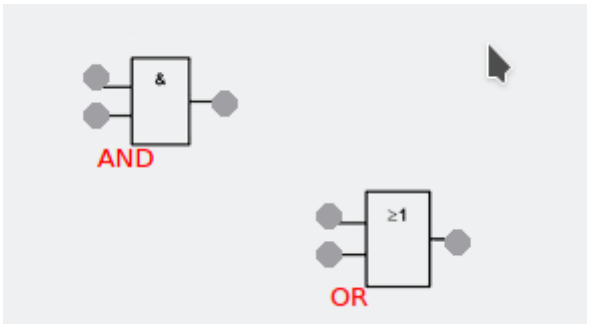


Рисунок 33 Размещение двух элементов

2. В левой части окна программы, найти кнопку «Соединить», нажать левой клавишей мыши.

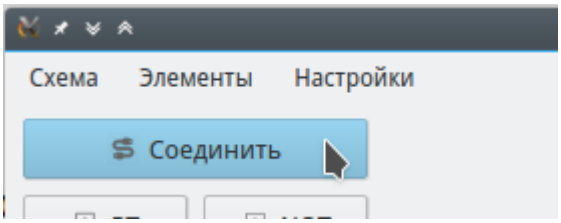


Рисунок 34 Выбор действия соединения

3. Выбрать контакт первого элемента. Когда контакт станет желтым, нажать левой кнопкой мыши. Если контакт выбран, то он будет подсвечен красной точкой.

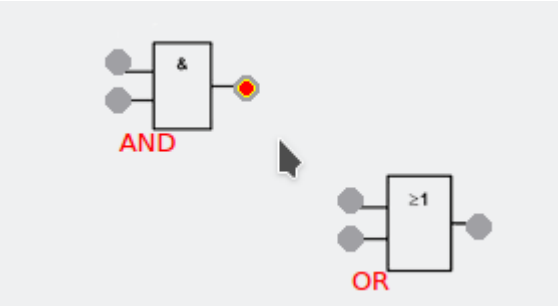


Рисунок 35 Выбран контакт первого элемента

4. Выбрать контакт второго элемента. Аналогично предыдущему пункту.

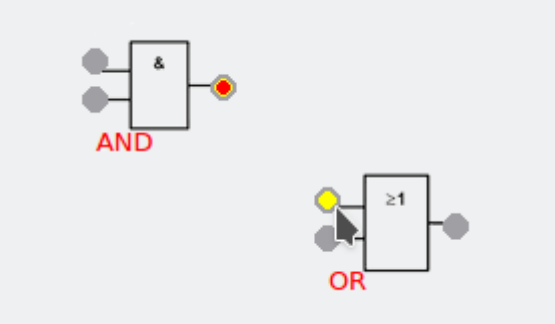


Рисунок 36 Выбран контакт второго элемента

5. После нажатия левой кнопкой мыши на втором контакте, в предыдущем пункте, должна появиться связь между двумя элементами.

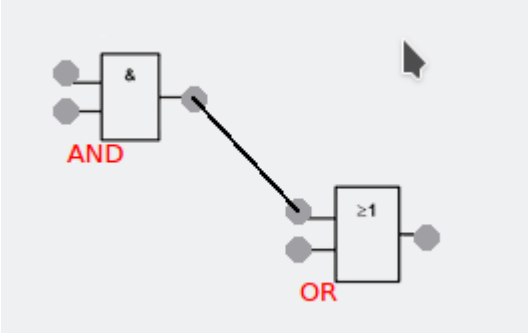


Рисунок 37 Создано соединение

Третий этап: когда расстановка элементов и их связывание выполнены, можно построить таблицу истинности. Для этого достаточно нажать кнопку Подсчитать в левой части окна программы.

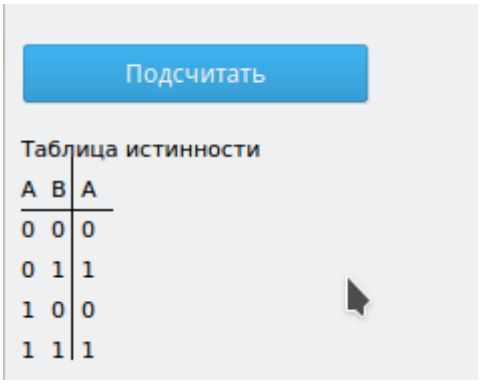


Рисунок 38 Кнопка подсчитать и таблица истинности

Четвертый этап: сохранение работы, готовой схемы. Для этого нужно зайти в пункт меню «Схема», выбрать «Сохранить в файл», выбрать имя файла и папку, куда сохранить. В любой момент, можно загрузить файл и продолжить конструирование схемы или переделать уже готовую.

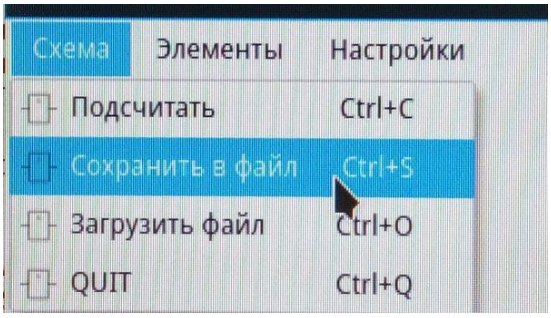


Рисунок 39 Меню программы

Пятый этап: удаление не нужных элементов выполняется нажатием правой кнопки мыши по самому элементу. Аналогично удаляется соединение: достаточно навести курсор на черную соединительную линию. Важно следить, чтобы в этот момент под линией не было другого элемента, так как он будет удален. Удаление элемента имеет больший приоритет, чем удаление соединения.

6. .Пакетный режим работы

Приложение поддерживает пакетный режим работы. Для этого достаточно при запуске указать два параметра: входной и выходной файлы. Входной файл это файл во внутреннем формате программы, а выходной это пустой файл формата txt. Он будет перезаписан таблицей истинности, которую сохранит приложение.

7. Техническая поддержка

При возникновении ошибок попробуйте закрыть программу и запустить её заново. Если ошибка повторяется, то напишите сообщение об ошибке и её описание на сайте проекта: https://github.com/VoidArray/AndORra

Раздел Issues. Потребуется регистрация. Автор ответит вам в течение двух рабочих дней.

Заключение

В мире происходит повсеместная автоматизация процессов. Это позволяет перейти на качественно более высокий уровень выполнения задач. В первую очередь, благодаря уменьшению механического труда и повышению производительности человека.

Выбранная область исследования основана на разделе математической логике и легко поддается автоматизации. Выполненное исследование позволило спроектировать и реализовать программу для конструирования и расчета комбинационных схем. Данное программное обеспечение может быть использовано в работе на небольших предприятиях, применяться студентами при проведения различных форм учебных занятий (дистанционных лекций, самостоятельной работы, совместной учебной деятельности студентов, консультаций).

Разработанный программный продукт позволяет легко его модернизировать, изменять под требования пользователя и дополнять благодаря продуманной архитектуре и открытому исходному коду. Программой может пользоваться, как начинающий постигать азы предметной области, так и человек, давно работающей в ней.

Литература

1. С. В. Фомин. Системы счисления. — М.: Наука, 1987. — 48 с. — (Популярные лекции по математике).

2. Доусон М. Программируем на Python. – СПб.: Питер, 2014. – 416 с

3.Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.

4. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.

5. Лутц М. Программирование на Python, том II, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.

6. Логика: учебник для юридических вузов / Под ред. В. П. Сальникова. — СПб.: Лексикон, 2001. — 320 с.

7. Прохоренок Н.А. Python 3 и PyQt. Разработка приложений. – СПб.: БХВ-Петербург, 2012. – 704 с.

8. Прохоренок Н.А. Python самое необходимое. — СПб.: БХВ-Петербург, 2011. — 416 с.

9. РосБизнесКонсалтинг: [сайт] // РБЦ - Москва. ведущая российская компания, работающая в сферах масс-медиа и информационных технологий. URL: rbc.ru (дата обращения: 14.04.2016)

10. ГОСТ 2.743-91. Обозначения условные графические в схемах. М ИПК.Издательство стандартов 2003 – 45 с.

11. Хахаев И.А. Практикум по алгоритмизации и программированию на Python. – М.: Альт Линукс, 2010. — 126 с.

12. Абачиев С.К. Формальная логика с элементами теории познания: учебник для вузов / С. К. Абачиев. — Ростов-на-Дону: Феникс, 2012. — 635 с.

13. Александров Д.Н. Логика. Риторика. Этика: учебное пособие / Д. Н. Александров. — М.: Флинта Наука, 2002. — 168 с.

14. Гетманова А. Д. Логика /А.Д. Гетманова. — М.: КноРус, 2012. — 416 с.

15. Гусев Д. А. Краткий курс логики. Искусство правильного мышления/Д.А. Гусев. — М.: НЦ ЭНАС, 2003. — 190 с.

16. Михайлов К.А. Логика. Практикум: учебное пособие для бакалавров / К. А. Михайлов, В. В. Горбатов. — М.: Юрайт, 2012. — 509 с.

17. Никифоров А. Л. Логика/А.Л. Никифоров. — М.: Весь мир, 2001. — 224 с.

18. Рузавин Г. И. Логика: Практический курс: учебник для вузов / Г. И. Рузавин. – М.: ЮНИТИ-ДАНА, 2002. — 256 с.

19. Демидов И.В. Логика: учебник / И. В. Демидов. — 8-е изд. – М.: Дашков и К, 2013. — 347 с.

20. Дмитревская И.В. Логика / И.В. Дмитревская. — М.: Флинта, 2013. — 384 с.

21. Ивин А. А. Логика: Учеб. пособие для вузов / А. А. Ивин. — М.: Высш. шк., 2004. — 304 с.

22. Ивлев Ю.В. Логика: учебник / Ю. В. Ивлев; Московский государственный университет им. М. В. Ломоносова. — 3-е изд., перераб. и доп. — Москва: Проспект, 2004. — 287 с.

23. Логика: Учебник для юридических вузов / под ред. проф. В. И. Кириллова. — Изд. 6-е, перераб. и доп. — М.: ТК Велби, Изд-во Проспект, 2008. — 240 с.

24. Ломиворотов М. М. Логика для юристов: Учеб. пособие в схемах и упражнениях/М.М Ломиворотов. — Волгоград, 2006. — 32 с.

25. Сковиков А.К. Логика: учебник и практикум для бакалавров / А.К. Сковиков. — Москва: Юрайт, 2013. — 575 с.

26. Тымцяс В. Г. Логика: Курс лекций / В. Г. Тымцяс. – М.: ПРИОР, 1999. — 160 с.

27. Челпанов Г. И. Учебник логики/Г.И. Челпанов. — М.: Прогресс, 1994. — 248 с.

28 Новиков Ю. В. Введение в цифровую схемотехнику. Курс лекций. — М.: Интернет-университет информационных технологий, 2006. — ISBN 5-94774-600-X

29 NIXP: [сайт] // NIXP - Москва. Русский проект об операционных системах, основанных на Unix URL: nixp.ru (дата обращения: 12.03.2016)

Приложения

Приложение 1.

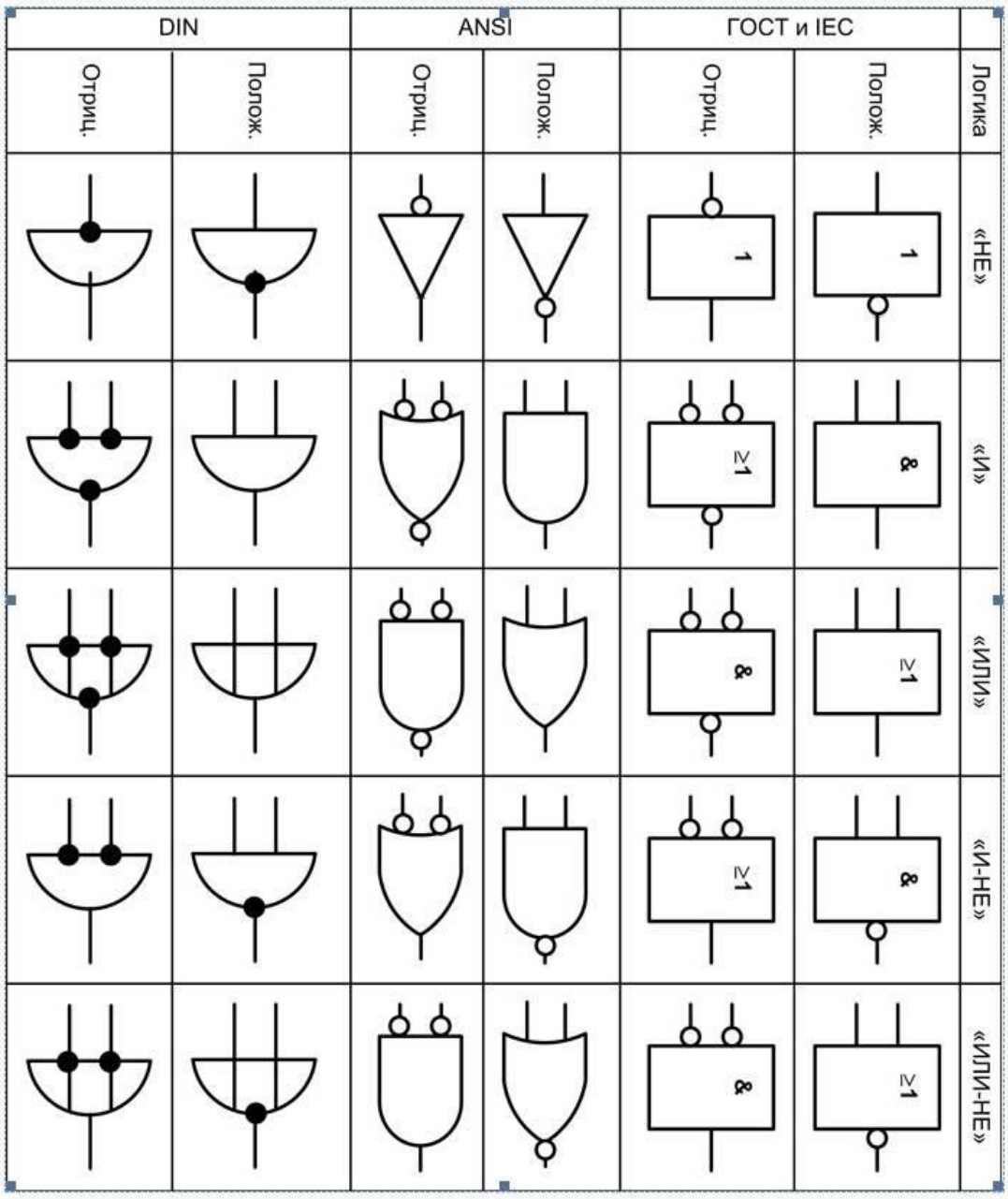


Рисунок 40 Сводная таблица логических элементов